Discrete Mathematics 309 (2009) 2305–2322

Contents lists available at ScienceDirect

Discrete Mathematics

journal homepage: www.elsevier.com/locate/disc

Connected domination of regular graphs☆

W. Duckworth[a,*], B. Mans[b]

[a] Mathematical Sciences Institute, The Australian National University, Canberra, ACT 0200, Australia

[b] Department of Computing, Macquarie University, Sydney, NSW 2109, Australia

ARTICLE INFO

Article history:
Received 23 November 2005
Received in revised form 1 May 2008
Accepted 9 May 2008
Available online 22 July 2008

Keywords:
Connected
Domination
Random
Regular
Graphs

ABSTRACT

A *dominating set* $\mathcal{D}$ of a graph G is a subset of $V(G)$ such that for every vertex $v \in V(G)$, either $v \in \mathcal{D}$ or there exists a vertex $u \in \mathcal{D}$ that is adjacent to v in G. Dominating sets of small cardinality are of interest. A *connected dominating set* $\mathcal{C}$ of a graph G is a dominating set of G such that the subgraph induced by the vertices of $\mathcal{C}$ in G is connected. A *weakly-connected dominating set* $\mathcal{W}$ of a graph G is a dominating set of G such that the subgraph consisting of $V(G)$ and all edges incident with vertices in $\mathcal{W}$ is connected. In this paper we present several algorithms for finding small connected dominating sets and small weakly-connected dominating sets of regular graphs. We analyse the average-case performance of these heuristics on random regular graphs using differential equations, thus giving upper bounds on the size of a smallest connected dominating set and the size of a smallest weakly-connected dominating set of random regular graphs.

1. Introduction

Throughout this paper we consider simple graphs (i.e. graphs with no loops or multiple edges) that are connected, undirected and unweighted. A graph G is said to be d-regular if every vertex in $V(G)$ has degree d (i.e. each vertex is adjacent to precisely d other vertices in G). When discussing any graph G, n denotes the cardinality of $V(G)$ and for d-regular graphs, we assume dn to be even. For other basic graph-theoretical definitions we refer the reader to [7].

A *dominating set* $\mathcal{D}$ of a graph G is a subset of $V(G)$ such that for every vertex $v \in V(G)$, either $v \in \mathcal{D}$ or there exists a vertex $u \in \mathcal{D}$ that is adjacent to v in G. Small dominating sets are of interest. The *domination number* of a graph G, denoted by $\gamma(G)$, is the size of a smallest dominating set of G.

A dominating set $\mathcal{I}$ of a graph G is said to be *independent* if no two vertices of $\mathcal{I}$ are connected by an edge of G. The *independent domination number* of a graph G, $\gamma_i(G)$, is the size of a smallest independent dominating set of G.

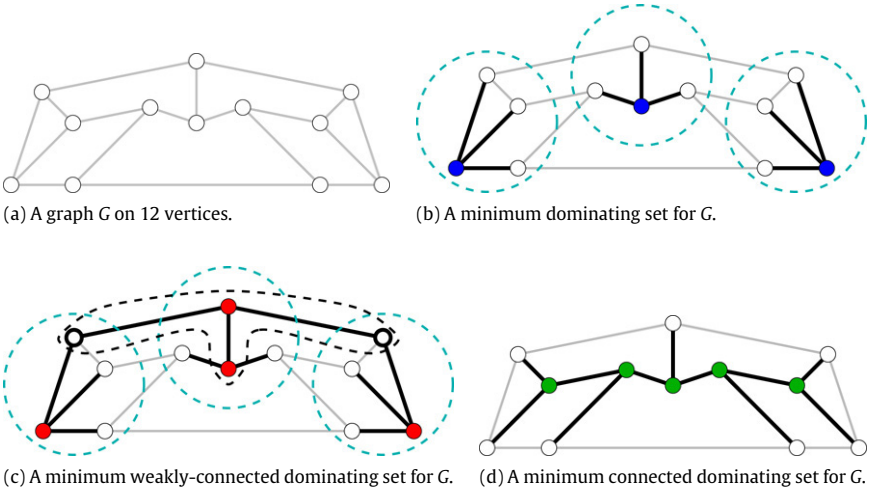
A *connected* dominating set $\mathcal{C}$ of a graph G is a dominating set such that the subgraph induced by the vertices of $\mathcal{C}$ in G is connected. The *connected domination number* of a graph G, denoted by $\gamma_c(G)$, is the size of a smallest connected dominating set of G.

Grossman [20] introduced another NP-hard variant of the minimum dominating set problem, that being the problem of finding a minimum *weakly-connected* dominating set. A weakly-connected dominating set $\mathcal{W}$ of a graph G is a dominating set such that the subgraph consisting of $V(G)$ and all edges incident with vertices in $\mathcal{W}$ is connected. The *weakly-connected domination number* of a graph G, denoted by $\gamma_w(G)$, is the size of a smallest weakly-connected dominating set of G. Small weakly-connected dominating sets have recently received much attention as they are of considerable interest for clustering mobile wireless ad hoc networks (see [6,8]).

☆ An abridged version of part of this work (with an alternate proof) appeared in the Proceedings of the 8th International Computing and Combinatorics Conference. Also, an abridged version of another part of this work appeared in the Proceedings of the 5th Conference on Algorithms and Complexity. This research was carried out whilst both authors were affiliated with the Department of Computing, Macquarie University, Sydney, NSW 2109, Australia.

* Corresponding author.
E-mail addresses: Billy.Duckworth@maths.anu.edu.au (W. Duckworth), bernard.mans@mq.edu.au (B. Mans).

0012-365X/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.disc.2008.05.029



(a) A graph G on 12 vertices. (b) A minimum dominating set for G.

(c) A minimum weakly-connected dominating set for G. (d) A minimum connected dominating set for G.

Fig. 1. Comparing different types of minimum dominating set.

We demonstrate the relationship between these different variants of dominating set using the small example given in Fig. 1. The graph in Fig. 1(a) is 3-regular and has twelve vertices, therefore, any dominating set must consist of at least three vertices. The darker solid (blue) vertices in Fig. 1(b) denote such a set and the dotted line surrounding each such vertex denotes the vertices it dominates. Note that dominating set in Fig. 1(b) is an *independent* dominating set (no edges exist between the vertices in the set). As every vertex not in the dominating set of Fig. 1(b) is dominated by precisely one vertex, this implies that a weakly-connected dominating set for this graph must contain at least four vertices. The solid darker (red) vertices in Fig. 1(c) form a weakly-connected dominating set of minimum size for this graph. White vertices with a dark rim are known as *gateways* in the field of mobile ad hoc networks. As the graph in Fig. 1 is 3-regular on twelve vertices, a minimum connected dominating set for this graph must consist of at least five vertices. The solid darker (green) vertices in Fig. 1(d) form a minimum connected dominating set for this graph.

The following section provides some known results about the problems defined above and also gives a summary of our results on the average-case performance of our algorithms. In Section 3 we describe a known model for generating regular graphs u.a.r. (uniformly at random) that we will use in our analysis. We also describe the notion of analysing the performance of algorithms on random graphs using systems of differential equations. In Section 4 we present and analyse algorithms for finding small connected dominating sets of regular graphs. In Section 5 we present and analyse algorithms for finding small weakly-connected dominating sets of regular graphs. The analysis of these algorithms uses differential equations and two theorems of Wormald [36,37].

2. Known bounds

2.1. $\gamma(G)$

For an arbitrary graph G the problem of determining $\gamma(G)$ is one of the core NP-hard optimisation problems in graph theory and this problem remains NP-hard even for planar graphs of maximum degree 3 (see Garey and Johnson [16]). As determining $\gamma(G)$ is a special instance of the minimum set cover problem, it is simple to deduce that for general graphs, $\gamma(G)$ is approximable within $1+\ln n$ using a result of Johnson [25]. Raz and Safra [31] showed that $\gamma(G)$ is not approximable within $c \log n$ for some $c > 0$. When restricted to graphs of bounded degree $d \geq 3$, Papadimitriou and Yannakakis [30] showed that the problem of determining $\gamma(G)$ remains NP-hard, is APX-complete and is approximable within $-\frac{1}{2} + \sum_{i=1}^{d+1} i^{-1}$.

2.2. $\gamma_i(G)$

Halldórsson [22] showed that for an arbitrary graph G, $\gamma_i(G)$ is not approximable within $n^{1-\epsilon}$ for any $\epsilon > 0$. Note that for a d-regular graph G, it is simple to verify that determining $\gamma_i(G)$ is approximable within $(d+1)/2$. For graphs with maximum degree d, Alimonti and Calamoneri [1] gave results for approximating $\gamma_i(G)$. Kann [26] showed that this problem is APX-complete for bounded degree graphs.

As we consider random d-regular graphs that are generated u.a.r., we require some associated notation. We say that a property $\mathcal{B} = \mathcal{B}_n$ of a random graph holds asymptotically almost surely (a.a.s.) if the probability that $\mathcal{B}$ holds tends to 1 as n tends to infinity. When d-regular graphs are the objects of consideration, this is modified so that n is restricted to even numbers if d is odd. For other basic random graph theory definitions we refer the reader to Bollobás [4].

Duckworth and Wormald [11] improved upon an earlier result of Molloy and Reed [29] by showing that for a random cubic (i.e. 3-regular) graph G, $\gamma_i(G)$ a.a.s. satisfies $0.2641n \leq \gamma_i(G) \leq 0.27942n$. Zito [38] presented upper and lower bounds on $\gamma_i(G)$ when G is a random d-regular graph and gave explicit values for these bounds when $3 \leq d \leq 7$. The lower bounds were, again, calculated by means of a direct expectation argument whilst the upper bounds were calculated by using differential equations to analyse the performance of a randomised algorithm that is based on repeatedly choosing vertices of a particular degree and deleting edges. (Note that for $d = 3$, the upper bound on $\gamma_i(G)$ in [38] is larger than the upper bound result presented in [11] and for $d > 3$, all upper bound results on $\gamma_i(G)$ in [38], when G is a random d-regular graph, have recently been improved by Duckworth and Wormald [13].)

2.3. $\gamma_c(G)$

For an arbitrary graph G, the problem of determining $\gamma_c(G)$ is a well-known NP-hard optimisation problem (see, for example, Haynes et al. [23]) and is polynomially equivalent to the *maximum leaf spanning tree* problem. (The non-leaf vertices of a spanning tree of a graph form a connected dominating set of the graph.) Define $\lambda(G)$ to be the maximum number of leaves in any spanning tree of G, so that for any graph G, $\lambda(G) = n - \gamma_c(G)$.

Solis-Oba [32] showed that the maximum leaf spanning tree problem is approximable within 2. Galbiati, Maffioli and Morzenti [15] showed that the same problem is not approximable within $1 + \epsilon$ for any $\epsilon > 0$ (unless P = NP).

Storer [33] showed that for every connected cubic graph G, $\lambda(G) \geq \lceil (n/4) + 2 \rceil$. Griggs, Kleitman and Shastri [18] showed that for every connected cubic graph G that has no subgraph isomorphic to "$K_4 - e$" (K_4 with one edge removed) $\lambda(G) \geq \lceil (n/4) + (4/3) \rceil$. Griggs and Wu [19] showed that for every connected graph G with minimum degree at least 4, $\lambda(G) \geq (2n + 8)/5$ and for every connected graph G with minimum degree at least 5, $\lambda(G) \geq (n + 4)/2$. For a connected graph G with minimum degree $k \geq 6$, the exact value of $\lambda(G)$ remains unknown. The results of [19,33] are essentially the best possible since there exist infinite d-regular graphs such that $\lambda(G) \leq \lceil (d - 2)n/(d + 1) \rceil + 2$. Loryś and Zwoźniak [27] showed that for cubic graphs, $\lambda(G)$ is approximable within 7/4. Duckworth and Wormald [12] gave a new derivation, at least to within an additive constant, of the main result of [33] and also showed that for every cubic graph G of girth at least 5, $\gamma_c(G) \leq 2n/3 + O(1)$. A linear programming technique that was developed in [12] also demonstrated the existence of infinitely many cubic graphs for which the algorithms only achieve these bounds.

Duckworth [9] showed that for a random cubic graph G, $\gamma_c(G)$ is a.a.s. less than $0.5854n$. This bound was achieved by using differential equations to analyse the performance of a randomised algorithm. Alon [2] proved by probabilistic methods that, for n sufficiently large, the size of a smallest *dominating set* of a graph with minimum degree d is at least $(1 + o_d(1))(1 + \ln(d + 1))n/(d + 1)$. Therefore, clearly, for such a graph G the same bound also holds for $\gamma_c(G)$. In fact, Caro et al. [5] showed that, for graphs of minimum degree d, the size of a minimum connected dominating set is essentially the same as the size of a minimum dominating set, for n and d sufficiently large.

For a d-regular graph G, a trivial lower bound on $\gamma_c(G)$ may be derived by considering the degrees of the vertices in the spanning tree that has a set of internal vertices of size $\gamma_c(G)$. Let $\mathcal{C}$ denote this set of internal vertices. Note that all vertices of $\mathcal{C}$ have degree at most d in the tree. All other vertices in the tree have degree 1 and there are $n - 1$ edges in the tree. This implies $d\mathcal{C} + n - \mathcal{C} \geq 2(n - 1)$ hence, $\gamma_c(G) \geq n/(d - 1)$ (asymptotically).

2.4. $\gamma_w(G)$

Chen and Liestman [6] introduced worst-case $(\ln \Delta + 1)$ approximation results for $\gamma_w(G)$, when G is a graph with bounded degree Δ. Their analysis techniques are similar to those used by Guha and Khuller [21] to bound $\gamma_c(G)$ when G is a bounded degree graph. Faster distributed algorithms have been recently introduced by Dubhashi et al. [8]. Clearly, for any graph G, $\gamma(G) \leq \gamma_w(G) \leq \gamma_c(G)$. Relationships between $\gamma(G)$, $\gamma_c(G)$ and $\gamma_w(G)$ have been extensively studied (see, for example, [2,5,14,23]).

Dunbar et al. [14] introduced the concept of having a weakly-connected *independent* dominating set, i.e. a weakly-connected dominating set $\mathcal{I}$ of a graph G such that no two vertices in $\mathcal{I}$ are connected by an edge of G. Define the minimum cardinality of all weakly-connected independent dominating sets of G as the *weakly-connected independent domination number* of G and denote this by $\gamma_w^i(G)$.

For arbitrary graphs, some relationships amongst the parameters $\gamma(G)$, $\gamma_w(G)$, $\gamma_c(G)$ and $\gamma_w^i(G)$ are known, for example, in [14], it was shown that

$$\gamma_w(G) \leq \gamma_c(G) \leq 2\gamma_w(G) - 1 \quad \text{and} \quad \gamma(G) \leq \gamma_w(G) \leq \gamma_w^i(G) \leq 2\gamma(G) - 1.$$

Caro et al. [5] showed that, for any graph G, of minimum degree d, that a.a.s.

$$\gamma_c(G) = \frac{(1 + o_d(1)) \ln(d + 1)}{d + 1} n. \tag{1}$$

This, along with a result of Alon [2] and a well-known result of Lovász [28], shows that for any graph G of minimum degree d, $\gamma(G)$ and $\gamma_c(G)$ are a.a.s. the same. As the weakly-connected dominating set returned by each of the algorithms we consider in Section 5 is actually a weakly-connected independent dominating set, our results also give upper bounds on $\gamma_w^i(G)$.

Table 1
Bounds on $\gamma_i(G)$, $\gamma_w(G)$ and $\gamma_c(G)$ for random d-regular n-vertex graphs

d	$\gamma_i(G)$	$\gamma_w(G)$	$\gamma_c(G)$	$\frac{\ln(d+1)}{d+1}n$
003	0.27942n	0.41198n	0.58542n	0.34657n
004	0.24399n	0.35861n	0.45651n	0.32189n
005	0.21852n	0.32051n	0.38607n	0.29863n
006	0.19895n	0.29136n	0.33935n	0.27799n
007	0.18329n	0.26806n	0.30520n	0.25993n
008	0.17037n	0.24890n	0.27874n	0.24414n
009	0.15948n	0.23277n	0.25743n	0.23026n
010	0.15015n	0.21896n	0.23978n	0.21799n
020	0.09830n	0.14243n	0.14932n	0.14498n
030	0.07526n	0.10850n	0.11210n	0.11077n
040	0.06181n	0.08873n	0.09010n	0.09057n
050	0.05285n	0.07559n	0.07715n	0.07709n
060	0.04640n	0.06614n	0.06730n	0.06739n
080	0.03764n	0.05335n	0.05407n	0.05425n
100	0.03190n	0.04500n	0.04550n	0.04569n

The columns in Table 1 summarise the results of this paper by giving our asymptotically almost sure upper bounds on $\gamma_w(G)$ and $\gamma_c(G)$ when G is a random d-regular graph on n vertices. For each d we also include asymptotically almost sure upper bounds on $\gamma_i(G)$ from [13] and the values $\ln(d+1)n/(d+1)$ (as a comparison to the results of Alon [2], Caro et al. [5] and Lovász [28]).

3. Random graphs and differential equations

The algorithms presented in subsequent sections are based on repeatedly choosing vertices of a particular current degree and deleting edges. At each iteration a different vertex is selected, a number of edges and/or vertices are removed from the graph and possibly one or more vertices are chosen to be part of the set that is under construction. In some algorithms, a priority is assigned to those vertices of a particular current degree. We refer to such algorithms as *prioritised*. The algorithms described in subsequent sections may be analysed using theorems of Wormald [34,36,37]. For each algorithm, this provides us with a set of differential equations whose solution describes the state of the algorithm during its execution. From this, we deduce asymptotically almost sure upper bounds on the size of the set of interest at the end of the algorithm.

As the analysis of the algorithms we present is carried out on random regular graphs, we give a brief overview of the model used for generating regular graphs u.a.r. The standard model for random d-regular graphs is as follows. Take a set of dn points in n buckets labelled $1, 2, \ldots, n$, with d points in each bucket, and choose u.a.r. a pairing $P = p_1, \ldots, p_{dn/2}$ of the points such that each p_i is an unordered pair of points and each point is in precisely one pair p_i. The resulting probability space of pairings is denoted by $\mathcal{P}_{n,d}$. Form a d-regular pseudograph on n vertices by placing an edge between vertices i and j for each pair in P having one point in bucket i and one in bucket j. In order to prove that a property is a.a.s. true of a uniformly distributed random d-regular (simple) graph, it is enough to prove that it is a.a.s. true of the pseudograph corresponding to a random pairing (for more information on this and other models of random regular graphs the reader is referred to see Bollobás [4] and Wormald [35]).

As in [36], we redefine this model by specifying that pairs are chosen sequentially. The first point in a pair may be selected using any rule, as long as the second point in that random pair is chosen u.a.r. from all the remaining unpaired points. This preserves the uniform distribution of the final pairing. When a pair has been determined in the sequential process, we say that it has been *exposed* and we say the graph *evolves* from all vertices starting out with degree zero to a graph in which all vertices have degree d. By exposing pairs in the order which an algorithm requests their existence, the generation of the random pairing may be combined with the algorithm (as in [3,10,34]). In this way, algorithms which delete edges may be described in terms of *operations* incorporated into the pairing generation. The definition of the operations may be extended to do whatever other tasks the algorithm needs to carry out.

The algorithm proper acts upon the final (pseudo)graph of the generation process. The set of exposed pairs builds up this final graph during the course of the generation process which incorporates the algorithm. The order in which the edges are deleted corresponds to the order in which the pairs were exposed. In what follows, we denote the set of vertices of degree i of the evolving graph, at time t, by $V_i = V_i(t)$, $0 \le i \le d$, and let $Y_i = Y_i(t)$ denote $|V_i|$. We may express the state of the evolving graph at any point during the execution of the algorithm by considering the variables Y_i. In order to analyse one of our randomised algorithms, we calculate the expected change in this state over one unit of time (a unit of time depends on the specific algorithm and is defined more clearly in subsequent sections) in relation to the expected change in the size of the set under construction. Let $D = D(t)$ denote the size of the set of interest at any stage of the algorithm (time t) and let $\mathbf{E}\Delta X$ denote the expected change in a random variable X conditional upon the history of the process. We then regard $\mathbf{E}\Delta Y_i/\mathbf{E}\Delta D$ as the derivative $\mathrm{d}Y_i/\mathrm{d}D$, which gives a system of differential equations. The solutions to these equations describe functions which represent the behaviour of the variables Y_i. There is a general result [34] which guarantees that the solutions

of the differential equations almost surely approximate the variables Y_i. The expected size of the set of interest may then be deduced.

4. Connected dominating set algorithms

The algorithms we describe are loosely based on the simple algorithm introduced by Guha and Khuller [21] that *grows* a spanning tree, T, of a graph, G. In this algorithm, vertices are repeatedly selected to be added to T based on their *colour*. As each vertex is added to T, edges may also be added to T and vertices may change colour. Three colours are used to colour the vertices; *black*, *white* and *grey*. Initially, all vertices are coloured white. At the end of the algorithm all vertices are either grey or black.

Start by choosing a white vertex v; colour v black and add v to T. Add the edges incident with v to T and colour the neighbours of v grey. The tree grows by repeatedly selecting grey vertices to add to T. At each iteration select a grey vertex v that has one or more white neighbours; add any edges that are incident with v to T, colour v black and colour the white neighbours of v grey. Once there are no white vertices, the algorithm terminates. Black vertices are internal vertices of T and grey vertices are external vertices (i.e. leaves) of T.

There are two key points to note about this algorithm. Firstly, edges that are incident with two grey vertices never becomes part of the tree and secondly, the input graphs may be assumed to be connected as this is a necessary requirement for the graph to have a connected spanning subgraph (or, indeed, a connected dominating set). These two points enable us to describe variations of this algorithm that construct small connected dominating sets of regular graphs (as opposed to growing trees) without the use of colours.

We first describe the above algorithm in terms of constructing a small connected dominating set of a regular graph by monitoring the degrees of the vertices at each step. The algorithm iteratively chooses vertices of a given degree for possible inclusion in the connected dominating set and then deletes its incident edges. The notion of colour may be adequately described in terms of the current degree of a vertex in the remainder of the graph to be processed.

At the start, all vertices have degree d and vertices of degree d represent non-dominated vertices. The initial step involves choosing the first vertex to add to the connected dominating set and deleting its incident edges. For each step, after the first, a vertex is selected for possible addition to the set from those vertices of degree greater than zero but less than d. Such a vertex will always exist, after the first step and before the completion of the algorithm, as the input graph is regular and is assumed to be connected. Choosing such a vertex at each iteration ensures the graph induced by the set of vertices returned is connected. Once such a vertex, u, is chosen, investigate the degree(s) of the neighbour(s) of u. If u has a neighbour of degree d, add u to the connected dominating set and delete the edges incident with u. If u has no neighbour of degree d, u and all its neighbours were already dominated at the start of the step. In which case we do not add u to the set, we just delete all edges incident with u and start a new step. At the end of the algorithm no vertices of degree d remain, ensuring that the set returned is dominating.

We now give a general description of the common features of the algorithms that we introduce in this section. For each algorithm a more detailed description is given in subsequent subsections. For a given algorithm we say that one step constitutes the process of selecting a vertex for possible inclusion in the connected dominating set and the deletion of a number of edges incident with that vertex.

For each of our algorithms the first step is identical. Select a vertex, u, u.a.r. from all the vertices of the input graph, add u to the connected dominating set and delete all edges incident with u. For each subsequent step, select a vertex, v, for possible inclusion in the connected dominating set u.a.r. from those vertices of a particular degree that is less than d but greater than zero. Once this selection has been made, select a given number of neighbours of v u.a.r. and investigate their degrees. If none of these neighbours has degree d, delete the edges incident with v and these selected neighbours. Otherwise, add v to the connected dominating set and delete all edges incident with v.

The algorithms we present vary in two ways; the subset of the vertices of degree less than d from which a vertex is selected at each step and the number of neighbours of the selected vertex that have their degrees investigated.

In the following subsections we give the full details (and analysis) of four greedy algorithms for finding a small connected dominating set of regular graphs. The algorithms are in increasing order of *greediness*. Parts of later algorithms may be the same as those of the previous algorithms, in which case, we do not reiterate the description and analysis.

4.1. Algorithm RAND_CDS

The first algorithm we consider for finding a small connected dominating set of a regular graph is a randomised version of the algorithm described above. In this algorithm we repeatedly choose vertices for possible inclusion in the connected dominating set at random. Each subsequent choice is made from those vertices that are eligible to become dominating set members so that the subgraph induced by the set constructed thus far remains connected. We call this algorithm RAND_CDS. The initial step selects the first connected dominating set vertex u.a.r. from all the vertices of the input graph and deletes its incident edges. For each subsequent step, select a vertex, u, for possible inclusion in the connected dominating set u.a.r. from all the vertices of current degree less than d but greater than zero. We add u to the connected dominating set if u has a neighbour of degree d. Otherwise, u and all its neighbours are already dominated. In either case, we delete all edges incident with u.

4.1.1. Combining the pairing process

The algorithm starts by selecting the first vertex of the connected dominating set u.a.r. and exposing its incident edges. This is achieved by selecting a mate for each free point in the bucket corresponding to the selected vertex. We say that the remainder of the algorithm proceeds in *operations*.

For each operation, we select a free point, p_1, u.a.r. from all the remaining free points in the buckets corresponding to the vertices of degree greater than zero. Using u to denote the vertex corresponding to the bucket that p_1 was selected from, we then expose all the remaining edges incident with u by selecting a mate for each free point in the bucket corresponding to u.

If all of the vertices represented by the buckets of the new neighbours of u now have degree greater than 1, both u and all its new neighbours were already dominated before these edges were exposed. In which case, the operation terminates without increasing the size of the connected dominating set. Otherwise, one or more of the new neighbours of u was not dominated before the edges were exposed and the operation is completed by adding u to the connected dominating set.

4.1.2. RAND_CDS analysis

The analysis of the performance of RAND_CDS is carried out using a system of differential equations. In order to achieve this, we first calculate the expected change in the variables Y_i, $0 \le i \le d-1$, and the expected change in the size of $\mathcal{C}$ for an operation. These equations are then used to form a system of differential equations.

For each edge exposed in the evolving graph two points are chosen. The first is chosen u.a.r. from a given set and the second is chosen u.a.r. from all the remaining free points. Let s denote the number of free points available in all buckets at a given stage (time t). Note that $s = s(t) = \sum_{i=0}^{d-1}(d-i)Y_i$. For our analysis it is convenient to assume that $s > \epsilon n$ for some arbitrarily small but fixed $\epsilon > 0$. Later, we discuss the last steps of the algorithm, where $s \le \epsilon n$.

The probability that, when selecting a free point u.a.r. from all the remaining free points (at time t), the point belongs to a vertex of degree j is P_j where

$$\mathrm{P}_j = \mathrm{P}_j(t) = \frac{(d-j)Y_j}{s}, \quad 0 \le j \le d-1.$$

The expected change in Y_i due to changing the degree of a vertex from i to $i+1$ by exposing an edge *to* it (at time t) is $\rho_i + o(1)$ where

$$\rho_i = \rho_i(t) = -\mathrm{P}_i + \mathrm{P}_{i-1}\delta_{i>0} + o(1), \quad 0 \le i \le 2$$

and for any statement S, δ_S evaluates to 1 if S is true and 0 otherwise. The term $o(1)$ comes about because the values of all these variables may change by a constant during the course of the operation being examined. Since $s > \epsilon n$ the error is in fact $O(1/n)$.

The probability that, when selecting the first free point in an operation u.a.r. from the vertices of degree greater than zero, the point belongs to a vertex of degree j is Q_j where

$$\mathrm{Q}_j = \mathrm{Q}_j(t) = \frac{(d-j)Y_j}{s - dY_0}, \quad 1 \le j \le d-1.$$

For an operation in which the first free point is selected u.a.r. from a vertex, u, of degree j, $1 \le j \le d-1$, the expected change in the variables Y_i, $0 \le i \le d-1$, is given by

$$-\delta_{i=j} + (d-j)\rho_i + o(1), \quad 0 \le i \le d-1.$$

The first term represents the removal of u from V_i (if $i = j$) and the second represents the change due to exposing $d-j$ edges incident with u.

For such an operation, the expected change in the size of the connected dominating set (at time t) is $1-(1-\mathrm{P}_0)^{d-j}+o(1)$. Note that we add u to the connected dominating with the probability that a new neighbour of u had degree zero at the start of the operation.

The expected change in Y_i when performing an operation (at time t) is then

$$\mathbf{E}\Delta Y_i = \mathbf{E}\Delta Y_i(t) = \sum_{j=1}^{d-1} \mathrm{Q}_j[-\delta_{i=j} + (d-j)\rho_i] + o(1), \quad 0 \le i \le d-1. \tag{2}$$

The expected change in the size of the connected dominating set when performing an operation (at time t) is $\mathbf{E}\Delta\mathcal{C} + o(1)$ where

$$\mathbf{E}\Delta\mathcal{C} = \mathbf{E}\Delta\mathcal{C}(t) = \sum_{j=1}^{d-1} 1 - (1-\mathrm{P}_0)^{d-j}. \tag{3}$$

The combined algorithm and pairing process is analysed using differential equations and in this way we prove the following theorem.

Table 2
Results for RAND_CDS

d	**RAND_CDS**	LB	d	**RAND_CDS**	LB
3	**0.7227**	0.5000	9	**0.3288**	0.1250
4	**0.5857**	0.3333	10	**0.3048**	0.1111
5	**0.4996**	0.2500	20	**0.1832**	0.0526
6	**0.4390**	0.2000	30	**0.1347**	0.0345
7	**0.3935**	0.1667	40	**0.1078**	0.0256
8	**0.3578**	0.1429	50	**0.0906**	0.0204

Theorem 1. *Let $d \geq 3$ be fixed. Then there exists a constant, c_1, given in Table 2, such that for a random d-regular graph on n vertices, the size of the connected dominating set returned by the algorithm RAND_CDS is a.a.s. at most $c_1 n + o(n)$.*

Proof. Eq. (2) representing the expected change in the variables Y_i for an operation forms the basis of a differential equation. Write $Y_i(t) = nz_i(t/n)$, $qs(t) = n\xi(t/n)$, $\mathrm{Q}_j(t) = n\bar{\mathrm{Q}}_j(t/n)$, $\mathrm{P}_i(t) = n\bar{\mathrm{P}}_i(t/n)$ and $\rho_i(t) = n\bar{\rho}_i(t/n)$.

The differential equation suggested is

$$z_i' = \sum_{j=1}^{d-1} \bar{\mathrm{Q}}_j[-\delta_{i=j} + (d-j)\bar{\rho}_i], \quad 0 \leq i \leq d-1. \tag{4}$$

Here differentiation is with respect to x and xn represents the number, t, of operations. From the definitions of s, P, Q and ρ, we have

$$\begin{aligned} \xi &= \sum_{i=0}^{d-1} (d-i)z_i, \\ \bar{\mathrm{P}}_j &= \frac{(d-j)z_j}{\xi}, \quad 0 \leq j \leq d-1, \\ \bar{\mathrm{Q}}_j &= \frac{(d-j)z_j}{\xi - dz_0}, \quad 1 \leq j \leq d-1, \\ \bar{\rho}_0 &= -\bar{\mathrm{P}}_0, \quad \text{and} \quad \bar{\rho}_i = \bar{\mathrm{P}}_{i-1} - \bar{\mathrm{P}}_i, \quad 1 \leq i \leq d-1. \end{aligned} \tag{5}$$

Eq. (3) representing the expected increase in $|\mathcal{C}| = C = C(t)$ for an operation and writing $C(t) = nz(t/n)$ suggests the differential equation for z as

$$z' = \sum_{j=1}^{d-1} 1 - (1 - \bar{\mathrm{P}}_0)^{d-j}. \tag{6}$$

The solution to this system of differential equations represents the cardinalities of the sets V_i and $\mathcal{C}$ (scaled by $1/n$) for given t. The initial conditions are $z_0(0) = 1$ and $z_i(0) = 0$ for $1 \leq i \leq d-1$.

We use a result from [36] to show that the functions representing the solutions to the differential equations almost surely approximate the variables Y_i/n and C/n with error $o(1)$.

For arbitrary small ϵ, define R, to be the set of all (t, z_i, z) for which $t > -\epsilon, \xi > \epsilon, z_0 > \epsilon, z > -\epsilon$ and $z_i < 1 + \epsilon$ where $0 \leq i \leq d-1$. Then, R defines a domain for the process so that [36, Theorem 6.1] may be applied. For part (i) of [36, Theorem 6.1] to hold, we must ensure that $Y_i(t)$ does not change too quickly throughout the process. This is immediate as we only consider asymptotics as $n \to \infty$ and, as d is assumed to be constant, only a constant number of edges are ever exposed in one operation.

Eqs. (2) and (3) verify part (ii) for a function λ_1 which goes to zero sufficiently slowly. Note in particular that since $\xi > \epsilon$ inside R, the assumption that $s > \epsilon n$ used in deriving these equations is justified. Part (iii) of [36, Theorem 6.1] is immediate from the form of the functions in Eqs. (2) and (3). The conclusion of [36, Theorem 6.1] therefore holds. This implies that the random variables Y_i/n and C/n a.a.s. remain within $o(1)$ of the corresponding deterministic solutions to the differential equations (4) and (6) until a point arbitrarily close to where it leaves R. We compute the ratio $\mathrm{d}z_i/\mathrm{d}z = z_i'(x)/z'(x)$ and we have

$$\frac{\mathrm{d}z_i}{\mathrm{d}z} = \sum_{j=1}^{d-1} \frac{\bar{\mathrm{Q}}_j[-\delta_{i=j} + (d-j)\bar{\rho}_i]}{1 - (1 - \bar{\mathrm{P}}_0)^{d-j}}, \quad 0 \leq i \leq d-1$$

where differentiation is with respect to z and all functions may be taken as functions of z.

By solving (numerically) this system of differential equations using a Runge–Kutta method, it was found that the solution hits a boundary of the domain at $z_0 = \epsilon$. Therefore, the solution of $z_0 = 0$ corresponds to the size of the connected dominating set when no vertices of degree zero remain. From the point after which [36, Theorem 6.1] does not apply until

the completion of the algorithm (i.e. $s < \epsilon n$), the change in each variable per step is bounded by a constant. Hence, letting ϵ tend to 0 sufficiently slowly, in $o(n)$ steps the change in the random variables Y_i and C is $o(n)$. □

For a few small values of d, Table 2 gives the constants, c_1, in Theorem 1 for which RAND_CDS returns a connected dominating set of a random d-regular graph of size at most $c_1 n + o(1)$ a.a.s. The lower bound (LB) is calculated using the trivial argument shown in Section 2.

4.2. Algorithm RAND_ONE_CDS

In the previous section, the algorithm RAND_CDS repeatedly selected vertices for possible inclusion in the connected dominating set u.a.r. from the vertices of current degree less than d but greater than zero. Once such a vertex had been chosen, the degrees of all its neighbours were investigated and all edges incident with the chosen vertex were deleted. Our second algorithm again selects vertices u.a.r. from the vertices of current degree less than d but greater than zero. This time, however, instead of investigating the degrees of all the neighbours of the selected vertex, we investigate the degree of just one neighbour selected u.a.r. We call this algorithm RAND_ONE_CDS. The rationale behind only investigating the degree of one neighbour of a selected vertex would be that should this vertex have degree less than d, the edge along which this investigation took place may be deleted as both its end-points are already dominated. This does not cause the size of the connected dominating set to increase and allows us to start a new step.

The initial step of this algorithm selects the first connected dominating set vertex u.a.r. from all the vertices of the input graph and deletes its incident edges. For each subsequent step, select a vertex, u, for possible inclusion in the connected dominating set u.a.r. from all the vertices of current degree less than d but greater than zero. Then, select a vertex, v, u.a.r. from the neighbours of u. If v has degree d, add u to the connected dominating set and delete all edges incident with u. Otherwise delete the edge between u and v.

4.2.1. Combining the pairing process

The algorithm starts by selecting the first vertex of the connected dominating set u.a.r. and exposing its incident edges. Again, we say that the remainder of the algorithm proceeds in *operations*.

For each operation we select a free point, p_1, u.a.r. from all the remaining free points in the buckets corresponding to the vertices of degree greater than zero and select a mate, p_2, for p_1 u.a.r. from all the remaining free points.

Using u and v to represent the vertices corresponding to the buckets that the points p_1 and p_2 belong to, this is equivalent to exposing an edge from u to v and we are then able to determine the degree of v. If v now has degree greater than 1, both u and v were already dominated before the edge was exposed and the operation terminates without increasing the size of the connected dominating set. Otherwise, v was not dominated before the edge was exposed and the operation is completed by adding u to the connected dominating set and exposing its remaining incident edges.

4.2.2. RAND_ONE_CDS analysis

The analysis of the performance of the algorithm RAND_ONE_CDS is carried out using a system of differential equations. In order to achieve this, we first calculate the expected change in the variables Y_i, $0 \le i \le d-1$, and the expected change in the size of $\mathcal{C}$ for an operation. These equations are then used to form a system of differential equations.

For an operation in which the first free point is selected u.a.r. from a vertex, u, of degree j, $1 \le j \le d-1$, the expected change in the variables Y_i, $0 \le i \le d-1$, is given by

$$-\delta_{i=j} + \rho_i + \mathrm{P}_0(d-j-1)\rho_i + (1-\mathrm{P}_0)\delta_{i=j+1} + o(1), \quad 0 \le i \le d-1.$$

The first term represents the removal of u from V_i (if $i = j$) and the second term represents the change due to exposing an edge by selecting the second point of the pair. With probability that the second point belonged to a vertex of degree zero, $d-j-1$ more edges are exposed giving the third term. With probability that the second point belonged to a vertex of degree greater than zero, u has its degree increased to $j+1$ giving the final term.

The expected change in Y_i when performing an operation (at time t) is $\mathbf{E}\Delta Y_i + o(1) = \mathbf{E}\Delta Y_i(t) + o(1)$ where $\mathbf{E}\Delta Y_i$ is given by

$$\sum_{j=1}^{d-1} \mathrm{Q}_j[-\delta_{i=j} + \rho_i + \mathrm{P}_0(d-j-1)\rho_i + (1-\mathrm{P}_0)\delta_{i=j+1}], \tag{7}$$

where $0 \le i \le d-1$ and Q_j remains the same as that defined earlier.

The expected change in the size of the connected dominating set when performing an operation (at time t) is $\mathbf{E}\Delta\mathcal{C} + o(1)$ where

$$\mathbf{E}\Delta\mathcal{C} = \mathbf{E}\Delta\mathcal{C}(t) = \mathrm{P}_0 \tag{8}$$

as we only add u to the connected dominating set if v had degree zero at the start of the operation.

The combined algorithm and pairing process is analysed using differential equations and in this way we prove the following theorem.

Table 3
Results for RAND_ONE_CDS

d	RAND_CDS	**RAND_ONE_CDS**	LB
3	0.7227	**0.6250**	0.5000
4	0.5857	**0.4900**	0.3333
5	0.4996	**0.4129**	0.2500
6	0.4390	**0.3612**	0.2000
7	0.3935	**0.3234**	0.1667
8	0.3578	**0.2942**	0.1429
9	0.3288	**0.2708**	0.1250
10	0.3048	**0.2515**	0.1111
20	0.1832	**0.1540**	0.0526
30	0.1347	**0.1148**	0.0345
40	0.1078	**0.0927**	0.0256
50	0.0906	**0.0784**	0.0204

Theorem 2. *Let $d \geq 3$ be fixed. Then there exists a constant, c_2, given in Table 3, such that for a random d-regular graph on n vertices, the size of the connected dominating set returned by the algorithm RAND_ONE_CDS is a.a.s. at most $c_2 n + o(n)$.*

Proof. Eq. (7) representing the expected change in the variables Y_i for an operation forms the basis of a differential equation. Write $Y_i(t) = nz_i(t/n)$, $s(t) = n\xi(t/n)$, $\mathrm{Q}_j(t) = n\bar{\mathrm{Q}}_j(t/n)$, $\mathrm{P}_i(t) = n\bar{\mathrm{P}}_i(t/n)$ and $\rho_i(t) = n\bar{\rho}_i(t/n)$. The differential equation suggested is z_i' which is given by

$$\sum_{j=1}^{d-1} \bar{\mathrm{Q}}_j[-\delta_{i=j} + \bar{\rho}_i + \bar{\mathrm{P}}_0(d-j-1)\bar{\rho}_i + (1-\bar{\mathrm{P}}_0)\delta_{i=j+1}], \quad 0 \leq i \leq d-1. \tag{9}$$

Here, differentiation is with respect to x and xn represents the number, t, of operations. The definitions of ξ, $\bar{\mathrm{P}}_j$, $\bar{\mathrm{Q}}_j$ and $\bar{\rho}_i$ remain the same as those in Eq. (5).

Eq. (8) representing the expected increase in $C = C(t) = |\mathcal{C}|$ for an operation and writing $C(t) = nz(t/n)$ suggests the differential equation

$$z' = \bar{\mathrm{P}}_0. \tag{10}$$

The solution to this system of differential equations represents the cardinalities of the sets V_i and $\mathcal{C}$ (scaled by $1/n$) for given t. The initial conditions are $z_0(0) = 1$ and $z_i(0) = 0$, $1 \leq i \leq d-1$.

By defining a suitable domain for the process, [36, Theorem 6.1] may be applied. The same arguments as those given for the analysis of the algorithm RAND_CDS in the previous section show that the conclusion of [36, Theorem 6.1] holds. This implies that the random variables Y_i/n and C/n a.a.s. remain within $o(1)$ of the corresponding deterministic solutions to the differential equations (9) and (10) until a point arbitrarily close to where it leaves the domain.

We compute the ratio $\mathrm{d}z_i/\mathrm{d}z = z_i'(x)/z'(x)$ and we have

$$\frac{\mathrm{d}z_i}{\mathrm{d}z} = \frac{1}{\bar{\mathrm{P}}_0} \sum_{j=1}^{d-1} \bar{\mathrm{Q}}_j[-\delta_{i=j} + \bar{\rho}_i + \bar{\mathrm{P}}_0(d-j-1)\bar{\rho}_i + (1-\bar{\mathrm{P}}_0)\delta_{i=j+1}], \quad 0 \leq i \leq d-1$$

where differentiation is with respect to z and all functions may be taken as functions of z. □

For a few small values of d, Table 3 gives the constants, c_2, in Theorem 2 for which RAND_ONE_CDS returns a connected dominating set of a random d-regular graph of size at most $c_2 n + o(1)$ a.a.s.

Both the algorithms presented thus far select each subsequent vertex for possible inclusion in the dominating set u.a.r. In the following sections we introduce algorithms that rely on the fact that vertices of a particular degree exist for a period of time.

4.3. Algorithm 1GREEDY_CDS

Note that, in the previous two algorithms, the number of vertices of degree $d-1$ after the first vertex is added to the connected dominating set is strictly greater than zero (a.a.s.). For an algorithm that repeatedly selects vertices of degree less than d for possible inclusion in the connected dominating set, each time investigating the degree of one neighbour, selecting a vertex of degree $d-1$ would give the largest expected number of newly dominated vertices. Also, for a period of time at least, doing this would generate new vertices of degree $d-1$ with positive probability. This gives rise to our third algorithm, 1GREEDY_CDS, which has a number of stages after the first step.

For the first stage, for each step, a vertex, u, of degree $d-1$ is selected u.a.r. and one neighbour, v, of u is selected u.a.r. to have its degree investigated. If v has degree d, we add u to the connected dominating set and delete all edges incident with u. Otherwise we just delete the edge between u and v. Once the number of vertices of degree $d-1$ reaches zero, the next stage of the algorithm commences in which u is not always selected from the vertices of degree $d-1$.

After the first stage, there are no vertices of degree $d-1$ and, assuming there exists vertices of degree d, the algorithm has not terminated. Once the next connected dominating set vertex is chosen (and an edge incident with a vertex of degree d is deleted), the number of vertices of degree $d-1$ is non-zero.

For any step in the second stage, we select vertices of degree $d-1$ when possible and delete all incident edges. The vertex is added to the connected dominating set if one of its neighbours had degree d before the edges were deleted. When no vertices of degree $d-1$ exist, a vertex u is chosen u.a.r. from those of maximum degree (less than d) and one of its neighbours, v, is selected u.a.r. Should v have degree d, u is added to the connected dominating set and all edges incident with u are deleted. If v has degree less than d, we delete the edge from u to v. Notice that this ensures that the maximum degree of the vertices of degree less than $d-1$ is decreasing as the algorithm proceeds. (A condition we will require in order to analyse this algorithm.)

4.3.1. Combining the pairing process

The algorithm 1GREEDY_CDS, for finding a small connected dominating set, $\mathcal{C}$, of random d-regular graphs, is combined with a pairing process that u.a.r. generates a random d-regular graph.

The first operation represents the process of selecting the first vertex of the connected dominating set and exposing its incident edges. After the first operation, we split the remainder of the algorithm into distinct ordered *phases*.

Phase k, $1 \leq k < d-1$, denotes the period of time from the first operation that selects a vertex u.a.r. from V_k up to but not including the first operation that selects a vertex u.a.r. from a vertex of degree larger than k. In Phase k, once the minimum non-zero degree of a vertex in the evolving graph is larger than k, the algorithm moves into Phase $k+1$ (or terminates if $k = d-1$).

In Phase 1, we select a vertex, u, u.a.r. from V_1 and expose an edge incident with u to a vertex v by selecting a free point from u and pairing this with a free point selected u.a.r. from all the remaining free points. If v had degree zero at the start of the operation we add u to the connected dominating set and expose the remaining edges incident with u. Otherwise we start a new operation. In Phase k, $2 \leq k \leq d-1$, if there exists a vertex of degree $d-1$, we select such a vertex, u, u.a.r. and expose all of its remaining incident edges. If any of the new neighbours of u had degree zero, we add u to the connected dominating set. If there are no vertices of degree $d-1$ when starting a new operation in Phase k, we select a vertex, u, u.a.r. from V_k and expose an edge incident with u. If the new neighbour of u now has degree 1, we add u to the connected dominating set and expose the remaining edges incident with u. Otherwise we start a new operation. In Phase k, once the minimum non-zero degree of a vertex in the evolving graph is larger than k, the algorithm moves into Phase $k+1$. At the end of Phase $d-1$, the algorithm terminates.

4.3.2. 1GREEDY_CDS analysis

The prioritised algorithm 1GREEDY_CDS *may* be analysed using [36, Theorem 6.1]. However, doing this requires checking complex conditions regarding derivatives. It also requires arguments involving branching processes and large deviation inequalities. It is much simpler to analyse the later phases of this algorithm using another theorem of Wormald [37, Theorem 1], which analyses deprioritised versions of prioritised algorithms.

For $d = 3$, the algorithm 1GREEDY_CDS is equivalent to the algorithm in [9] that finds a small connected dominating set of cubic graphs. The algorithm in [9] is analysed as follows. Letting variables Y_i ($i = 0, \ldots, 3$) denote the number of vertices of current degree i, the expected values of Y_i are estimated throughout the algorithm for each i using differential equations. It is shown that with high probability the variables are concentrated near their expected values. The analysis in [11] has complications arising from the fact that priority is given to vertices currently of a given degree.

In Phase 1 of the algorithm 1GREEDY_CDS, all operations start by selecting a vertex of degree 1 from the evolving graph. In Phase k, $k > 1$, there are a mixture of operations. In particular, each operation that selects a vertex of degree k is followed by a number (possibly zero) of operations that start by selecting a vertex of degree 1.

In [9], the sets of operations which start with an operation that selects a vertex of degree k and all subsequent operations that start by selecting a vertex of degree 1 are referred to as *clutches*. The concept of a clutch of operations is also utilised in [37, Theorem 1].

The setting of [37, Theorem 1] concerns a class of processes applied to the random pairing. As described above, this may be defined in terms of the generation algorithm which exposes pairs. The beginning of the generation algorithm is the empty pairing G_0. The pairing G_{t+1} is obtained from G_t by applying an operation which may expose some of the pairs; the *degree* of a bucket is the number of points it contains in exposed pairs. The operation, op_t, which is applied to G_t must be one of some pre-specified set of operations, Op_i, $i = 1, \ldots, d$, where Op_i consists of selecting a bucket u of degree i (vertex of degree $d-i$) in G_t u.a.r., and then applying some specified set of tasks, resulting in G_{t+1}. A subset C of $V(G) \cup E(G)$ is selected during the operations, with $C_0 = \emptyset$ initially, and $C = C_t$ for the pairing G_t. As in the previous sections, for $1 \leq i \leq d$, let $Y_i = Y_i(t)$ denote the number of buckets of degree i in G_t, and let $Y_{d+1} = Y_{d+1}(t)$ denote cardinality of the set C_t.

The combined algorithm and pairing process is analysed using differential equations and in this way we prove the following theorem.

Theorem 3. *Let $d \geq 3$ be fixed. Then there exists a constant, c_3, given in Table 4, such that for a random d-regular graph on n vertices, the size of the connected dominating set returned by the algorithm 1GREEDY_CDS is a.a.s. at most $c_3 n + o(n)$.*

Table 4
Results for 1GREEDY_CDS

d	RAND_CDS	RAND_ONE_CDS	1GREEDY	LB
3	0.7227	0.6250	**0.5854**	0.5000
4	0.5857	0.4900	**0.4575**	0.3333
5	0.4996	0.4129	**0.3880**	0.2500
6	0.4390	0.3612	**0.3420**	0.2000
7	0.3935	0.3234	**0.3085**	0.1667
8	0.3578	0.2942	**0.2825**	0.1429
9	0.3288	0.2708	**0.2616**	0.1250
10	0.3048	0.2515	**0.2443**	0.1111
20	0.1832	0.1540	**0.1552**	0.0526
30	0.1347	0.1148	**0.1182**	0.0345
40	0.1078	0.0927	**0.0970**	0.0256
50	0.0906	0.0784	**0.0830**	0.0204

Proof. We may verify the hypotheses of [37, Theorem 1].

The expected change in Y_i when performing an Op_1 in Phase 1 (at time t) is $f_{i,1}$ where

$$f_{i,1} = -\delta_{i=1} + \rho_i + \mathrm{P}_0(d-2)\rho_i + (1-\mathrm{P}_0)\delta_{i=2}, \quad 0 \le i \le d-1. \tag{11}$$

The first term represents the removal of u from V_1 (if $i = 1$) and the second term represents the change due exposing the first edge to v. With probability v had degree zero at the start of the operation, we expose $d-2$ more edges giving the third term. With probability v already had degree greater than zero at the start of the operation, u has its degree increased to 2 giving the final term.

The expected change in the size of the connected dominating set when performing an operation in Phase 1 (at time t) is $f_{d+1,1}$ where

$$f_{d+1,1} = \mathrm{P}_0 \tag{12}$$

as we only add u to the set if the first edge is exposed to a vertex of degree 0.

In Phase k, $2 \le k \le d-1$, we have two types of operation. The expected change in Y_i when performing an operation of Type Op_1 in Phase k (at time t) is $f_{i,1}$ where

$$f_{i,1} = -\delta_{i=1} + (d-1)\rho_i, \quad 0 \le i \le d-1,$$

representing the removal of u from V_1 and exposing its remaining $d-1$ incident edges.

The expected change in Y_i when performing an Op_k in Phase k (at time t) is $f_{i,k}$ where

$$f_{i,k} - \delta_{i=k} + \rho_i + \mathrm{P}_0(d-k-1)\rho_i + (1-\mathrm{P}_0)\delta_{i=k+1}, \quad 0 \le i \le d-1.$$

The expected change in the size of the connected dominating set when performing an Op_1 in Phase k is the probability that when the $d-1$ edges are exposed, a vertex of degree zero has its degree increased to 1. This is given by $1-(1-\mathrm{P}_0)^{d-1}+o(1)$.

The expected change in the size of the connected dominating set when performing an Op_k in Phase k is the probability that the first edge is exposed to a vertex of degree zero and this is $\mathrm{P}_0 + o(1)$.

Hypothesis (i) of [37, Theorem 1] is immediate since in any operation at most $d-1$ edges are exposed and the size of $\mathcal{C}$ increases by at most one. The functions $f_{i,r}$ satisfy (ii) because from the equations $f_{i,r}$ defined above, their (possible) singularities satisfy $s = 0$, which lies outside $\mathcal{D}_\epsilon$ since in $\mathcal{D}_\epsilon$, $s \ge y_d \ge \epsilon$. Hypothesis (iii) follows from the equations $f_{i,r}$ again using $s \ge y_d \ge \epsilon$ and the boundedness of the functions y_i (which follows from the boundedness of $\mathcal{D}_\epsilon$).

It turns out that these hold for each d in Table 4, and that in each case m differs at the point where z_0 becomes zero. For ϵ sufficiently small, the value of $\tilde{y}_{d+1}(x_m)$ may be computed numerically (the result is shown in Table 4), and then by [37, Theorem 1], this is the asymptotic value of the size of the connected dominating set $\mathcal{C}$ at the end of some randomised algorithm. So the conclusion is that a random d-regular graph a.a.s. has a connected dominating set of size at most $n\tilde{y}_{d+1}(x_m) + o(n)$. □

For a few small values of d, Table 4 gives the constants, c_3, in Theorem 3, for which 1GREEDY_CDS returns a connected dominating set of a random d-regular graph of size at most $c_3 n + o(1)$ a.a.s. We compare these results with those for the algorithms RAND_CDS and RAND_ONE_CDS and against the lower bound (LB). Note that for larger d, the algorithm 1GREEDY_CDS performs worse that RAND_ONE_CDS.

4.4. Algorithm kGREEDY_CDS

During the second stage of the algorithm 1GREEDY_CDS, if there exists a vertex of degree $d-1$, such a vertex is selected u.a.r. and all of its neighbours have their degree investigated. This is an attempt to control the number of vertices of maximum degree (less than $d-1$). This may also be achieved without investigating all the neighbours of a vertex of degree $d-1$.

Table 5
Results for *k*GREEDY_CDS

d	RAND_CDS	RAND_ONE_CDS	1GREEDY	*k*GREEDY	LB
3	0.7227	0.6250	0.5854	**0.5854**	0.5000
4	0.5857	0.4900	0.4575	**0.4565**	0.3333
5	0.4996	0.4129	0.3880	**0.3860**	0.2500
6	0.4390	0.3612	0.3420	**0.3393**	0.2000
7	0.3935	0.3234	0.3085	**0.3051**	0.1667
8	0.3578	0.2942	0.2825	**0.2787**	0.1429
9	0.3288	0.2708	0.2616	**0.2573**	0.1250
10	0.3048	0.2515	0.2443	**0.2397**	0.1111
20	0.1832	0.1540	0.1552	**0.1493**	0.0526
30	0.1347	0.1148	0.1182	**0.1121**	0.0345
40	0.1078	0.0927	0.0970	**0.0910**	0.0256
50	0.0906	0.0784	0.0830	**0.0771**	0.0204

Our final connected dominating set algorithm, *k*GREEDY_CDS, again, has two stages. The first stage is the same as that for 1GREEDY_CDS. In the second stage, should there exist a vertex of degree $d-1$, we select such a vertex, u, u.a.r. and investigate the degrees of $k-1$ of its neighbours selected u.a.r, where k represents the maximum degree (less than $d-1$) of the vertices in the remainder of the graph to be processed. If any of these neighbours has degree d, u is added to the connected dominating set and all edges incident with u are deleted. If none of these neighbours have degree d, then u and all those neighbours selected were already dominated at the start of the step and we delete the edges incident with u and those $k-1$ neighbours. On the other hand, when no vertices of degree $d-1$ exist, a vertex, u, is chosen u.a.r. from all vertices of maximum degree less than d and one of its neighbours has its degree investigated. This controls the maximum degree amongst the vertices of degree less than $d-1$ that are still to be processed.

4.4.1. Combining the pairing process

The first operation represents the process of selecting the first vertex of the connected dominating set and exposing its remaining incident edges. Each operation after the first is denoted by one iteration of a while loop which involves u.a.r. selecting a vertex of given degree, exposing one or more edges incident with this vertex, the possible addition of the vertex to the connected dominating set and possibly exposing more edges.

After the first operation, we split the remainder of the algorithm into $d-1$ distinct ordered phases, Phase 1, Phase 2, . . ., Phase $d-1$. A Type k operation, $1 \le k \le d-1$, refers to an operation in which the first vertex chosen in the operation is chosen u.a.r. from V_k. We informally define Phase k as the period of time from the first Type k operation up to but not including the first Type k' operation where $k' > k$. In Phase $k < d-1$, once Y_k reaches zero, the algorithm moves into Phase $k+1$.

In Phase 1, we select a vertex, u, u.a.r. from V_1 and expose an edge incident with u to a vertex v. If v had degree zero at the start of the operation, we add u to the connected dominating set and expose the remaining edges incident with u. Otherwise we start a new operation.

In Phase k, $2 \le k \le d-1$, if there exists a vertex of degree 1, we select such a vertex, u, u.a.r. and expose $k-1$ of its remaining incident edges. If any of the new neighbours of u had degree zero, we add u to the connected dominating set and expose its remaining incident edges. If there are no vertices of degree 1, we select a vertex, u, u.a.r. from V_k and expose an edge incident with u. If the new neighbour of u now has degree 1, we add u to $\mathcal{C}$ and expose the remaining edges incident with u. Otherwise we start a new operation.

4.4.2. kGREEDY_CDS analysis

The combined algorithm and pairing process is analysed using differential equations and in this way we prove the following theorem.

Theorem 4. *Let $d \ge 3$ be fixed. Then there exists a constant, c_4, given in Table 5, such that for a random d-regular graph on n vertices, the size of the connected dominating set returned by the algorithm k GREEDY_CDS is a.a.s. at most $c_4 n + o(n)$.*

Proof. Phase 1 is the same as that for the algorithm 1GREEDY_CDS and therefore the equations giving the expected change in the variables Y_i and the expected increase in the size of $\mathcal{C}$ for an Op_1 in Phase 1 are the same as those given for an operation in the previous section. These are Eqs. (11) and (12) respectively.

The expected change in Y_i when performing an Op_1 in Phase k, $2 \le k \le d-1$, (at time t) is $f_{i,1}$ where

$$f_{i,1} = -\delta_{i=1} + (k-1)\rho_i + (1-(1-\mathrm{P}_0)^{k-1})(d-k)\rho_i + (1-\mathrm{P}_0)\delta_{i=k}, \quad 0 \le i \le d-1.$$

The expected change in Y_i when performing an Op_k in Phase k (at time t) is $f_{i,k}$ where

$$f_{i,k} = -\delta_{i=k} + \rho_i + \mathrm{P}_0(d-k-1)\rho_i + (1-\mathrm{P}_0)^{k-1}\delta_{i=k+1}, \quad 0 \le i \le d-1.$$

The expected change in the size of the connected dominating set when performing an Op_1 in Phase k is the probability that we hit a vertex of degree zero when the $k-1$ edges are exposed. This is $1-(1-\mathrm{P}_0)^{k-1}+o(1)$.

The expected change in the size of the connected dominating set when performing an Op_k in Phase k is the probability that we hit a vertex of degree zero when the first edge is exposed and this is $\mathrm{P}_0+o(1)$.

The same arguments as those given by the analysis of 1GREEDY_CDS in the previous section show that [37, Theorem 1] may be applied to the process. The conclusion of [37, Theorem 1] therefore holds implying that the random variables Y_i/n and C/n a.a.s. remain within $o(1)$ of the corresponding deterministic solutions to the differential equations until a point arbitrarily close to where they leave the domain. □

For a few small values of d, Table 5 gives the constants, c_4, in Theorem 4.

5. Weakly-connected dominating set algorithms

5.1. Growing a weakly-connected component

Finding a small weakly-connected dominating set $\mathcal{W}$ of an arbitrary connected graph may be easily achieved by growing a weakly-connected component. Three colours are used to colour the vertices; *black*, *white* and *grey*. Initially, all vertices are coloured white. Start by choosing a white vertex v; colour v black and add v to $\mathcal{W}$. Colour the neighbours of v grey. The component grows by repeatedly selecting a grey vertex v that has one or more white neighbours. Select a white neighbour, say w, colour it black and colour the white neighbours of w grey. Once there are no white vertices, the algorithm terminates.

The heuristic we describe is similar to this algorithm and adapted for regular graphs. It is a randomised greedy algorithm that is based on repeatedly selecting vertices of given current degree from an ever-shrinking subgraph of the input graph. At the start of our algorithm, all vertices have degree d. Throughout the execution of our algorithm, vertices are repeatedly chosen at random from a given set. A neighbour of such a vertex may be selected for inclusion in the set under construction; deleting edges at each iteration.

For a d-regular graph, G, the algorithm constructs a subset, $\mathcal{W}$, of the vertices of G in a series of *steps*. Each step starts by selecting a vertex u.a.r. from those vertices of a particular current degree. The first step is unique in the sense that it is the only step in which a vertex is selected u.a.r. from the vertices of degree d. We select such a vertex, u, u.a.r. from all the vertices of the input graph to add to $\mathcal{W}$. We then delete all edges incident with u.

For each step after the first, we select a vertex, u, u.a.r. from those vertices of positive degree that is less than d. Such a vertex will always exist (after the first step and before the completion of the algorithm) as the input graph is assumed to be connected. We then select a neighbour, v, of u u.a.r. and investigate its degree. If v has degree d, we add v to $\mathcal{W}$ and delete all edges incident with v. Otherwise, (v has degree less than d) we simply delete the edge between u and v and start another step (without adding a vertex to $\mathcal{W}$).

At any given stage of the algorithm we say that the *component* represents the set $\mathcal{W}$ constructed thus far, along with all edges of G that are incident to vertices in $\mathcal{W}$. Every vertex of $\mathcal{W}$ is chosen from the vertices of degree d and all edges that are deleted are either incident with a vertex in $\mathcal{W}$ or have both end-points of degree less than d. This ensures that once no vertices of degree d remain $\mathcal{W}$ is a dominating set, in fact, it is a weakly-connected independent dominating set. We say that the component *grows* as edges and vertices are added to it.

5.2. A randomised greedy algorithm

The component starts out as a copy of the complete bipartite graph $K_{1,d}$. This is achieved by selecting the first vertex of $\mathcal{W}$ u.a.r. from all the vertices of the input graph. Pseudo-code for our algorithm, is given in Fig. 2. In the algorithm, $N(u)$ denotes the set of vertices incident to u in G.

At each iteration, the algorithm adds to the component either an edge between two vertices that are already in the component or a new vertex that is incident with at least one vertex that is already in the component. In the latter instance all edges incident with the new vertex are also added to the component (along with any of its neighbours that are not already part of the component).

5.3. Average-case analysis

As with the previous algorithms, the combined algorithm and pairing process is analysed using differential equations and in this way we prove the following theorem.

Theorem 5. *Let $d \geq 3$ be fixed. Then there exists a constant, c_5, given in Table 6, such that for a random d-regular graph on n vertices, the size of the weakly-connected dominating set returned by the algorithm RAND_GREEDY is a.a.s. at most $c_5 n + o(n)$.*

Proof. Denote each iteration of the while loop in Fig. 2 as one *operation*. In order to analyse the algorithm we calculate the expected change in the variables Y_i in relation to the expected change in $W = |\mathcal{W}|$ for an operation. These equations are then used to formulate a differential equation.

```
Input  :  A d-regular n-vertex graph, G.
Output :  A weakly-connected dominating set W for G.

1  W ← ∅;
2  select u u.a.r. from the vertices of degree d;
3  W ← {u};
4  delete all edges incident with u;
5  while (there are vertices of degree d remaining)
   {
6    select u u.a.r. from the vertices of positive degree less than d;
         \\Random greedy selection criteria
7    select v u.a.r. from N(u);
         \\Attempt to increase the number of vertices in the component
8    delete the edge between u and v;
9    if (v has degree d − 1) { W ← W ∪ {v};
                               delete all edges incident with v; }
   }
```

Fig. 2. Algorithm Rand_Greedy.

Table 6
Bounds on $\gamma_w(G)$ for a random d-regular graph, G

d	$\alpha(\gamma_w(G))$	$\beta(\gamma_w(G))$	$\frac{n\ln(d+1)}{d+1}$	d	$\alpha(\gamma_w(G))$	$\beta(\gamma_w(G))$	$\frac{n\ln(d+1)}{d+1}$
03	0.4120n	0.4120n	0.3466n	09	0.2852n	0.2328n	0.2303n
04	0.3586n	0.3586n	0.2780n	10	0.2659n	0.2190n	0.2180n
05	0.4167n	0.3205n	0.2986n	20	0.1657n	0.1424n	0.1450n
06	0.3713n	0.2914n	0.2780n	40	0.1005n	0.0887n	0.0906n
07	0.3362n	0.2681n	0.2599n	60	0.0741n	0.0661n	0.0674n
08	0.3081n	0.2489n	0.2441n	80	0.0593n	0.0533n	0.0543n

Note that (depending on the algorithm being analysed), it *may* be necessary to calculate the expected change in the variables Y_i in relation to the expected change in W for all $0 \le i \le d$. However, as our algorithm terminates when $Y_0 = 0$, computing the expected change in the variable Y_0 in relation to the expected change in W may be sufficient (providing that, suitable equations may be derived to represent this process that do not involve the variables Y_j, $1 \le j \le d$). It will become apparent that this is the case. These equations may then be used to formulate a differential equation.

Let $s = s(t)$ denote the number of free points in the evolving graph G at a given stage (time t). Recall that $s = \sum_{i=0}^{d-1}(d-i)Y_i$. For our analysis it is convenient to assume that $s > \epsilon n$ for some arbitrarily small but fixed $\epsilon > 0$. Operations when $s \le \epsilon n$ will be discussed later.

For each operation, we select a vertex, u, u.a.r. from $V(G)\backslash\{V_0 \cup V_d\}$ and expose u.a.r. an edge to a vertex v. If v now has degree 1, all edges incident with v are exposed. Otherwise, only the edge incident with u and v is exposed. The expected change in Y_0 due to decreasing the degree of v is $-dY_0/s$. Decreasing the degree of u by one has no effect on Y_0 as $u \notin V_0$. In the event v had degree 0, a further change in V_0 may result if any of the other new neighbours of v had degree 0. The expected number of neighbours of v that had degree 0, given that v had degree 0 and $u \notin V_0$ is $d(d-1)Y_0/s$. Therefore, the expected change in Y_0 when performing an operation is $\mathbf{E}\Delta Y_d + o(1) = \mathbf{E}\Delta Y_d(t) + o(1)$ where

$$\mathbf{E}\Delta Y_d = \frac{-dY_0}{s}\left(1 + \frac{d(d-1)Y_0}{s}\right). \tag{13}$$

The expected change in the size of $\mathcal{W}$ when performing an operation is $\mathbf{E}\Delta W + o(1) = \mathbf{E}\Delta W(t) + o(1)$ which is simply given by

$$\mathbf{E}\Delta W = \frac{dY_0}{s}. \tag{14}$$

The $o(1)$ terms in Eqs. (13) and (14) are due to the fact that the values of all the variables may change by a constant during the course of the operation being examined. Since $s > \epsilon n$ the error is in fact $O(1/n)$.

We use (13) and (14) to formulate a differential equation.

Write $Y_i(t) = nz_i(t/n)$, $W(t) = nz(t/n)$ and $s(t) = n\xi(t/n)$. From the definition of s we have

$$\xi = \sum_{i=0}^{d-1}(d-i)z_i.$$

Eq. (13) representing the expected change in Y_0 for an operation forms the basis of a differential equation. The differential equation suggested is

$$\frac{\delta z_0}{\delta x} = \frac{-dz_0}{\xi}\left(1 + \frac{d(d-1)z_0}{\xi}\right), \tag{15}$$

where $x = t/n$ and t is the number of operations.

Eq. (14) representing the expected increase in the size of $\mathcal{W}$ for an operation suggests the differential equation for z as

$$\frac{\delta z}{\delta x} = \frac{dz_0}{\xi}. \tag{16}$$

We compute the ratio $\delta z/\delta z_0$, and we have

$$\frac{\delta z}{\delta z_0} = \frac{-1}{1 + \frac{d(d-1)z_0}{\xi}}. \tag{17}$$

Notice that with every edge exposed, ξ decreases by 2. It follows that

$$\frac{\delta \xi}{\delta z_0} = \frac{2\xi}{dz_0}.$$

Solving this equation with initial condition $\xi = d$ when $z_0 = 1$ gives $\xi = dz_0^{2/d}$. Substituting this expression for ξ into Eq. (17), we have

$$\frac{\delta z}{\delta z_0} = \frac{-1}{1 + (d-1)z_0^{(\frac{d-2}{d})}}. \tag{18}$$

The solution to this differential equation represents the cardinalities of V_0 and $\mathcal{W}$ (scaled by $1/n$) for given t up until $\xi = \epsilon$. After which point, the change in the variables per operation is bounded by a constant and the error in the solution is $o(n)$.

Using the substitution $(d-1)^{\frac{1}{d-2}} = z_0^{1/d}$ we have

$$\begin{aligned} z &= -d(d-1)^{\frac{-d}{d-2}} \int \frac{w^{d-1}}{1+w^{d-2}}\,\delta w \\ &= -d(d-1)^{\frac{-d}{d-2}} \int w\,\delta w + d(d-1)^{\frac{-d}{d-2}} \int \frac{w}{1+w^{d-2}}\delta w. \end{aligned} \tag{19}$$

The second integral in Eq. (19) is a known indefinite integral (see [17, Equation (2.146)]) so we have

$$\begin{aligned} z = {} & \frac{2d(d-1)^{\frac{-d}{d-2}}}{d-2} \sum_{k=1}^{\lfloor \frac{d-2}{2} \rfloor} \sin\left(2\frac{\Pi(2k-1)}{d-2}\right) \arctan\left(\frac{w - \cos\left(\frac{\Pi(2k-1)}{d-2}\right)}{\sin\left(\frac{\Pi(2k-1)}{d-2}\right)}\right) \\ & - \frac{d(d-1)^{\frac{-d}{d-2}}}{d-2} \sum_{k=1}^{\lfloor \frac{d-2}{2} \rfloor} \cos\left(2\frac{\Pi(2k-1)}{d-2}\right) \ln\left(1 - 2w\cos\left(\frac{\Pi(2k-1)}{d-2}\right) + w^2\right) \\ & - \frac{d(d-1)^{\frac{-d}{d-2}} \ln(1+w)}{d-2}\,(d \bmod 2) - \frac{d(d-1)^{\frac{-d}{d-2}} w^2}{2} + C. \end{aligned}$$

Substituting $w = (d-1)^{\frac{1}{d-2}}$ to find C and substituting $w = 0$ to find the end of the process we find that $\gamma_w(G)/n$ is at most

$$\begin{aligned} & \frac{d}{2(d-1)} + \frac{d(d-1)^{\frac{-d}{d-2}} \ln(1 + (d-1)^{\frac{1}{d-2}})}{d-2}\,(d \bmod 2) - \frac{2d(d-1)^{\frac{-d}{d-2}} \sum_{k=1}^{\lfloor \frac{d-2}{2} \rfloor} \sin\left(2\frac{(2k-1)\Pi}{d-2}\right) \arctan\left(\frac{\cos(\frac{(2k-1)\Pi}{d-2})}{\sin\left(\frac{(2k-1)\Pi}{d-2}\right)}\right)}{d-2} \\ & + \frac{2d(d-1)^{\frac{-d}{d-2}} \sum_{k=1}^{\lfloor \frac{d-2}{2} \rfloor} \sin\left(2\frac{(2k-1)\Pi}{d-2}\right) \arctan\left(\frac{-(d-1)^{\frac{1}{d-2}} - \cos\left(\frac{(2k-1)\Pi}{d-2}\right)}{\sin\left(\frac{(2k-1)\Pi}{d-2}\right)}\right)}{d-2} + \frac{d(d-1)^{\frac{-d}{d-2}}}{d-2} \\ & \times \sum_{k=1}^{\lfloor \frac{d-2}{2} \rfloor} \cos\left(2\frac{(2k-1)\Pi}{d-2}\right) \ln\left(1 - 2(d-1)^{\frac{1}{d-2}} \cos\left(\frac{(2k-1)\Pi}{d-2}\right) + (d-1)^{\frac{2}{d-2}}\right). \end{aligned}$$

Finally for $d = 3$ we note that the above equation simplifies to $\gamma_w(G) \leq 3\ln(3)n/8$ and for $d = 4$ it simplifies to $\gamma_w(G) \leq 2(3 - \ln(4))n/9$. This completes the proof of Theorem 5. □

We now approximate the solution to Eq. (19) for values of d larger than 4. We do so by lower bounding the function $w/(1 + w^{d-2})$ in the interval

$$0 \leq w \leq (d-1)^{\frac{1}{d-2}}.$$

Note that over this interval for w, the function $w/(1 + w^{d-2})$ is always positive and in this interval, its derivative equals zero at just one point. This implies,

$$\frac{w}{1+w^{d-2}} \leq \frac{(d-3)^{\left(\frac{d-3}{d-2}\right)}}{d-2}, \quad 0 \leq w \leq (d-1)^{\frac{1}{d-2}}.$$

When $w/(1 + w^{d-2}) = (d-3)^{\left(\frac{d-3}{d-2}\right)}/(d-2)$ we have $w = (d-3)^{\frac{-1}{d-2}}$. Note that

$$0 \leq (d-3)^{\frac{-1}{d-2}} \leq (d-1)^{\frac{1}{d-2}}$$

for the values of d under consideration.

We compute two linear functions of w. We show that the first of these functions, in the interval

$$0 \leq w \leq (d-3)^{\frac{-1}{d-2}},$$

is at most

$$\frac{w}{1+w^{d-2}}$$

and the other, in the interval

$$(d-3)^{\frac{-1}{d-2}} \leq w \leq (d-1)^{\frac{1}{d-2}},$$

is also at most

$$\frac{w}{1+w^{d-2}}.$$

Lemma 6. *For every $d > 4$ in the interval $0 \leq w \leq (d-3)^{\frac{-1}{d-2}}$,*

$$\frac{d-3}{d-2}w \leq \frac{w}{1+w^{d-2}}.$$

Proof. Rearrange the expression above to get

$$(d-3)w(1+w^{d-2}) \leq (d-2)w$$
$$w^{d-2} \leq \frac{1}{d-3},$$

which completes the proof as $w \leq (d-3)^{\frac{-1}{d-2}}$. □

Lemma 7. *For every $d > 4$ in the interval $(d-3)^{\frac{-1}{d-2}} \leq w \leq (d-1)^{\frac{1}{d-2}}$*

$$\frac{(d-3)\left((d-1)^{\left(\frac{1}{d-2}\right)} - w\right)}{2-d+(d-2)(d-1)^{\left(\frac{1}{d-2}\right)}, (d-3)^{\left(\frac{1}{d-2}\right)}} \leq \frac{w}{1+w^{d-2}}.$$

Proof. Rearrange the expression above to get

$$(d-3)(d-1)^{\left(\frac{1}{d-2}\right)} \leq (d-2)(d-1)^{\left(\frac{1}{d-2}\right)}(d-3)^{\left(\frac{1}{d-2}\right)}w + (d-3)w^{d-1} - w - (d-3)(d-1)^{\left(\frac{1}{d-2}\right)}w^{d-2}.$$

As w and $(d-3)(d-1)^{\left(\frac{1}{d-2}\right)}w^{d-2}$ are positive:

$$(d-3)(d-1)^{\left(\frac{1}{d-2}\right)} \leq (d-2)(d-1)^{\left(\frac{1}{d-2}\right)}(d-3)^{\left(\frac{1}{d-2}\right)}w + (d-3)w^{d-1}.$$

As $0 \leq w \leq (d-1)^{\frac{1}{d-2}}$:

$$(d-3)(d-1)^{\left(\frac{1}{d-2}\right)} \leq (d-2)(d-1)^{\left(\frac{1}{d-2}\right)}(d-3)^{\left(\frac{1}{d-2}\right)}(d-1)^{\left(\frac{1}{d-2}\right)} + (d-3)(d-1)^{\left(\frac{d-1}{d-2}\right)}.$$

As $(d-3)^{\left(\frac{1}{d-2}\right)} \leq d-3$

$$(d-3)(d-1)^{\left(\frac{1}{d-2}\right)} \leq (d-2)(d-1)^{\left(\frac{1}{d-2}\right)}(d-3)(d-1)^{\left(\frac{1}{d-2}\right)} + (d-3)(d-1)^{\left(\frac{d-1}{d-2}\right)}$$
$$-1 \leq (d-1)^{\left(\frac{1}{d-2}\right)}$$

which completes the proof as $(d-1)^{\left(\frac{1}{d-2}\right)} \geq 0$. □

We have

$$\begin{aligned} z &= \frac{d}{2(d-1)} - d(d-1)^{\frac{-d}{d-2}} \int_0^{(d-1)^{\frac{1}{d-2}}} \frac{w}{1+w^{d-2}} \delta w \\ &\leq \frac{d}{2(d-1)} - d(d-1)^{\frac{-d}{d-2}} \int_0^{(d-3)^{\frac{-1}{d-2}}} \left(\frac{d-3}{d-2} w\right) \delta w \\ &\quad - d(d-1)^{\frac{-d}{d-2}} \int_{(d-3)^{\frac{-1}{d-2}}}^{(d-1)^{\frac{1}{d-2}}} \frac{(d-3)\left((d-1)^{\left(\frac{1}{d-2}\right)} - w\right)}{2-d+(d-2)(d-1)^{\left(\frac{1}{d-2}\right)}(d-3)^{\left(\frac{1}{d-2}\right)}} \delta w. \end{aligned}$$

Evaluating this enables us to prove that for a random d-regular graph on n vertices, $(d > 4)$, $\gamma_w(G)$ is a.a.s. less than

$$\left(\frac{d}{2(d-1)} - \frac{d(d-3)^{\frac{d-3}{d-2}}}{2(d-2)(d-1)^{\frac{d-1}{d-2}}}\right) n + o(n).$$

In Table 6, we present our upper bounds on $\gamma_w(G)$, $\alpha(\gamma_w(G))$, the exact solution to Eq. (17), $\beta(\gamma_w(G))$, (produced by using a Runge–Kutta method) along with the value $n \ln(d+1)/(d+1)$ from Eq. (1) as a comparison to the known asymptotic results for $\gamma(G)$ and $\gamma_c(G)$.

5.4. Comparison with degree-greedy heuristics

Having analysed what seems to be the simplest algorithm for finding a small weakly-connected dominating set of regular graphs, it is natural to consider whether different heuristics may give an improved result. The so-called *degree-greedy* algorithms (like those in the previous sections) that are based on choosing a vertex of a particular degree at each iteration give improved results for various other problems on regular graphs.

There are several degree-greedy algorithms that one may design for finding a small weakly-connected dominating set of a regular graph. Providing all of these algorithms are based on iteratively growing a single weakly-connected component, these may only differ in two ways; namely, for each iteration (after the initial operation), the type of greedy selection criteria used and by how many vertices the set is allowed to increase per iteration. These are represented by the lines 6 and 7 in the while loop of the heuristic presented in Fig. 2.

The remaining features remain the same: the initial operation (lines 1–4) must choose a vertex of degree 0 (deleting some or all of its incident edges) to initiate the growing component. It should also be clear that, in order to ensure only *one* component is "grown", each iteration must start by selecting a vertex of degree greater than 0 (line 6). As vertices of degree 0 represent non-dominated vertices, any such algorithm may terminate once the number of vertices of degree 0 in the graph reaches 0.

To analyse such a heuristic, using the differential equation technique as we have, it is usually necessary to develop equations based on the expected changes in the variables Y_i, however, as the algorithm may terminate once no vertices of degree 0 remain, their incident edges exposed, it is sufficient to track the variable Y_0 as opposed to the vector $(Y_1, Y_2, \ldots, Y_d)$.

As any other alternative heuristics can only modify lines 6 and 7 (and the vertex, u, selected in each iteration must have degree larger than 0), it is therefore immediate that any such selection may not affect the variable Y_0. The choice of which vertex of degree larger than 0 to choose is therefore immaterial in this regard.

Once u has been selected, Y_0 may only decrease if edges incident with u are exposed. It is well known that, for a random regular graph, the neighbourhood of a vertex, up to a constant distance, is a.a.s. acyclic, (see, for example, [24]) and as we are only interested in the degrees of vertices at distance at most 2 from u, the subgraph considered in each iteration will a.a.s. be a tree.

It may be observed that investigating the degree of one neighbour of u per iteration or investigating more than one neighbour of u per iteration will have no effect on the resulting differential equation. The latter may be seen as an algorithm

that performs a sequence of operations per iteration. The first in the sequence selects a vertex u from those of a given degree and investigates the degree of one of its neighbours. In each remaining operation in the sequence, the same vertex u is selected and one more of its neighbours has its degree investigated. (This may be achieved by a standard modification to the pairing process which generates the graph u.a.r.; each point in a pair is chosen sequentially where the first point may be chosen by any rule. It is known that this preserves the uniformity of the final pairing.) As soon as the required number of neighbours have had their degree investigated, the sequence terminates. Each neighbour of u of degree 0 encountered must be included in the set under construction. (Again, this comes from the assumption that vertices of degree larger than 0 are dominated and as once each of these vertices has its degree investigated, an edge incident edge with that vertex is deleted). All edges incident with these vertices would then be exposed.

It is not difficult to see that this would mean that the performance of algorithms that base each selection on choosing vertices of minimum (or maximum) degree and algorithms that iteratively choose one (or more than one) neighbours per iteration, would be represented by Eq. (17), and therefore will have the same average-case performance (as the solution to the differential equation would, of course, be the same).

References

[1] P. Alimonti, T. Calamoneri, Improved approximations of independent dominating set in bounded degree graphs, in: Proceedings of the 22nd International Workshop on Graph-Theoretic Concepts in Computer Science, in: Lecture Notes in Computer Science, vol. 1197, Springer-Verlag, 1997, pp. 2–16.
[2] N. Alon, Transversal numbers of uniform hypergraphs, Graphs and Combinatorics 6 (1) (1990) 1–4.
[3] M. Beis, W. Duckworth, M. Zito, Large k-independent sets of random regular graphs, Electronic Notes in Discrete Mathematics 19 (2005) 321–327.
[4] B. Bollobás, Random Graphs, Academic Press Incorporated, Harcourt Brace Jovanovich, London, 1985.
[5] T. Caro, D.B. West, R. Yuster, Connected domination and spanning trees with many leaves, SIAM Journal on Discrete Mathematics 13 (2) (2000) 202–211.
[6] Y.P. Chen, A.L. Liestman, Approximating minimum size weakly-connected dominating sets for clustering mobile ad-hoc networks, in: Proceedings of the 3rd ACM International Symposium on Mobile Ad-Hoc Networking and Computing, ACM press, 2002, pp. 165–172.
[7] R. Diestel, Graph Theory, Springer-Verlag, New York, 2000.
[8] D. Dubhashi, A. Mei, A. Panconesi, J. Radhakrishnan, A. Srinivasan, Fast distributed algorithms for (weakly) connected dominating sets and linear-size skeletons, in: Proceedings of the 14th ACM–SIAM Symposium on Discrete Algorithms, ACM Press, 2003, pp. 717–724.
[9] W. Duckworth, Minimum connected dominating sets of random cubic graphs, The Electronic Journal of Combinatorics 9 (1) (2002) #R7(electronic).
[10] W. Duckworth, B. Mans, Randomised greedy algorithms for finding small k-dominating sets of random regular graphs, Random Structures and Algorithms 27 (3) (2005) 401–412.
[11] W. Duckworth, N.C. Wormald, Minimum independent dominating sets of random cubic graphs, Random Structures and Algorithms 21 (2) (2002) 147–161.
[12] W. Duckworth, N.C. Wormald, Linear programming and the worst-case analysis of greedy algorithms on cubic graphs (submitted for publication).
[13] W. Duckworth, N.C. Wormald, On the independent domination number of random regular graphs, Combinatorics, Probability and Computing 15 (4) (2006) 513–522.
[14] J.E. Dunbar, J.W. Grossman, J.H. Hattingh, S.T. Hedetniemi, A.A. McRae, On weakly-connected domination in graphs, Discrete Mathematics 167–168 (1997) 261–269.
[15] G. Galbiati, F. Maffioli, A. Morzenti, A short note on the approximability of the maximum leaves spanning tree problem, Information Processing Letters 52 (1) (1994) 45–49.
[16] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman and Company, San Francisco, California, 1979.
[17] I.S. Gradshteyn, I.M. Ryzhik, Table of Integrals, Series, and Products, Academic Press, San Diego, California, USA, 1996.
[18] J.R. Griggs, D.J. Kleitman, A. Shastri, Spanning trees with many leaves in cubic graphs, Journal of Graph Theory 13 (6) (1989) 669–695.
[19] J.R. Griggs, M. Wu, Spanning trees in graphs of minimum degree 4 or 5, Discrete Mathematics 104 (2) (1992) 167–183.
[20] J.W. Grossman, Dominating sets whose closed stars form spanning trees, Discrete Mathematics 169 (1–3) (1997) 83–94.
[21] S. Guha, S. Khuller, Approximation algorithms for connected dominating sets, Algorithmica 20 (1998) 374–387.
[22] M.M. Halldórsson, Approximating the minimum maximal independence number, Information Processing Letters 46 (4) (1993) 169–172.
[23] T.W. Haynes, S.T. Hedetniemi, P.J. Slater, Domination in Graphs: Advanced Topics, Marcel Dekker, New York, 1998.
[24] S. Janson, T. Łuczak, A. Rucinski, Random Graphs, Wiley, New York, 2000.
[25] D.S. Johnson, Approximation algorithms for combinatorial problems, Journal of Computer and System Sciences 9 (1974) 256–278.
[26] V. Kann, On the approximability of NP-complete optimisation problems, Ph.D. Thesis, Department of Numerical Analysis, Royal Institute of Technology, Stockholm, 1992.
[27] K. Loryś, G. Zwoźniak, Approximation algorithm for the maximum leaf spanning tree problem for cubic graphs, in: Proceedings of the 10th Annual European Symposium on Algorithms, in: Lecture Notes in Computer Science, vol. 2461, Springer-Verlag, 2002, pp. 686–697.
[28] L. Lovász, On the ratio of optimal integral and fractional covers, Discrete Mathematics 13 (4) (1975) 383–390.
[29] M. Molloy, B. Reed, The dominating number of a random cubic graph, Random Structures & Algorithms 7 (3) (1995) 209–221.
[30] C.H. Papadimitriou, M. Yannakakis, Optimization, approximation and complexity classes, Journal of Computer and System Sciences 43 (3) (1991) 425–440.
[31] R. Raz, S. Safra, A sub-constant error-probability low-degree test and a sub-constant error-probability PCP characterization of NP, in: Proceedings of the 29th Annual ACM Symposium on the Theory of Computing, ACM Press, 1997, pp. 475–484.
[32] R. Solis-Oba, 2-approximation algorithm for finding a spanning tree with maximum number of leaves, in: Proceedings of the 6th Annual European Symposium on Algorithms, in: Lecture Notes in Computer Science, vol. 1461, Springer-Verlag, 1998, pp. 441–452.
[33] J.A. Storer, Constructing full spanning trees for cubic graphs, Information Processing Letters 13 (1) (1981) 8–11.
[34] N.C. Wormald, Differential equations for random processes and random graphs, Annals of Applied Probability 5 (1995) 1217–1235.
[35] N.C. Wormald, Models of random regular graphs, in: London Mathematical Society Lecture Note Series, vol. 267, 1999, pp. 239–298.
[36] N.C. Wormald, The differential equation method for random graph processes and greedy algorithms, in: Lectures on Approximation and Randomized Algorithms, PWN, Warsaw, 1999, pp. 73–155.
[37] N.C. Wormald, Analysis of greedy algorithms on graphs with bounded degrees, Discrete Mathematics 273 (1–3) (2003) 235–260.
[38] M. Zito, Greedy algorithms for minimisation problems in random regular graphs, in: Proceedings of the 9th Annual European Symposium on Algorithms, in: Lecture Notes in Computer Science, vol. 2161, Springer-Verlag, 2001, pp. 524–536.